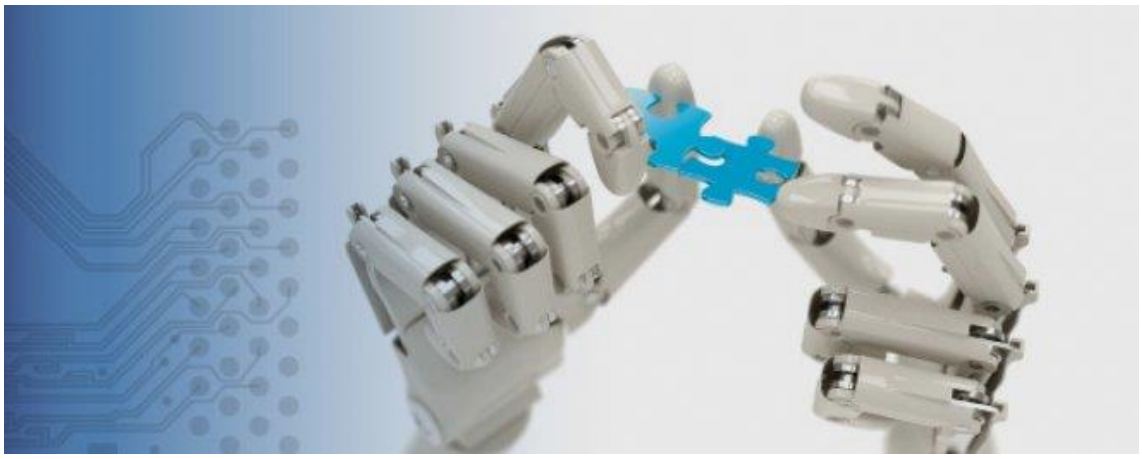
ROBOTICS MASTER

Universitat de Vic



Subject: Perception Systems

Session1: Sensors and Measurements

Exercixse 1.4: Webcam – Find value of a pixel, Force pixels

Author: Toni Guasch Serra

Date: 2015-10-29

Exercise 1.4.

Using a program to capture images on-line from your webcam (Exercise 1.4 of Robotics Integration):

- Try to find the value of the central pixel currently captured by the camera.
- Change that pixel, as well as its 8 immediate neighbors according whatever idea you have.

hint: http://docs.opencv.org/modules/core/doc/basic_structures.html#mat

---

Exercise 1.4.

Using a program to capture images on-line from your webcam (Exercise 1.4 of Robotics Integration):

A. Try to find the value of the central pixel currently captured by the camera.
B. Change that pixel, as well as its 8 immediate neighbors according whatever idea you have. hint: http://docs.opencv.org/modules/core/doc/basic_structures.html#mat

Both parts of the exercise done;

- Find the RGB values of a central pixel of an image
- Force this central pixel and 2 levels of neighbors (8 pixels) to a constant value.

Observations;

- This exercise is linked to exercise 1.4 of Modul4: Robotics Integration. Where the webcam_capture code was made and executed to see the image of the webcam
- Due that the mentioned exercise is not working fine, this exercise neither has been tested and executed
- Camera is not working in VMware were I have installed Ubuntu
- The exercise is done looking the reference documentation and searching examples in Google. The program was modified to cover the requirements, as if it were to build and execute
- Could contain small programming errors since it has not been build. But this is the proposed solution to the required modification of the program

Actions:

- This code will be tested and improved in class PC's with an external USB camera
- Will be revised the settings of VMware to see if the laptop camera resource is working fine in Ubuntu and if it is accessible. Search a solution to test my own codes

Error:

```
toni@ubuntu:~/Documents/GIT/Webcam$ cd webcam_capture
toni@ubuntu:~/Documents/GIT/Webcam/webcam_capture$ ls
build           INSTRUCTIONS_INSTALL.txt  README.md  test.txt
CMakeLists.txt  LICENSE                   src
toni@ubuntu:~/Documents/GIT/Webcam/webcam_capture$ cd build
toni@ubuntu:~/Documents/GIT/Webcam/webcam_capture/build$ ./webcam_capture
Opening video device 0
Error opening the camera. May be invalid device id. EXIT program.
toni@ubuntu:~/Documents/GIT/Webcam/webcam_capture/build$
```

Code:

Modifications marked in bold

```
#include "cv.h"
#include "highgui.h"
#include <iostream>
#include <cstdlib>

int main(int argc, char *argv[])
{
        //OpenCV video capture object
  cv::VideoCapture camera;

        //OpenCV image object
  cv::Mat image;

        //camera id . Associated to device number in /dev/videoX
        int cam_id;

        //check user args
        switch(argc)
        {
                case 1: //no argument provided, so try /dev/video0
                        cam_id = 0;
                        break;
                case 2: //an argument is provided. Get it and set cam_id
                        cam_id = atoi(argv[1]);
                        break;
                default:
                        std::cout << "Invalid number of arguments. Call program as: webcam_capture [video_device_id]. " << std::endl;
                        std::cout << "EXIT program." << std::endl;
                        break;
        }

        //advertising to the user
        std::cout << "Opening video device " << cam_id << std::endl;

  //open the video stream and make sure it's opened
  if( !camera.open(cam_id) )
        {
    std::cout << "Error opening the camera. May be invalid device id. EXIT program." << std::endl;
    return -1;
  }

  //capture loop. Out of user press a key
  while(1)
        {
                //Read image and check it
    if(!camera.read(image))
                {
      std::cout << "No frame" << std::endl;
      cv::waitKey();
    }

//-----------------------------------------------------------------------------------------
//CENTRAL PIXEL
//-----------------------------------------------------------------------------------------
//Obtain the value of the central pixel of the image

//Central pixel?
        int row_central = image.rows * 0.5;
        int col_central = image.cols * 0.5;
```

```
//Get value of each channel of RGB
        int RED_Value = image.at<cv::Vec3b>(row_central,col_central)[0] ;
        int GREEN_Value = image.at<cv::Vec3b>(row_central,col_central)[1] ;
        int BLUE_Value = image.at<cv::Vec3b>(row_central,col_central)[2] ;


//Print value RGB components (Central pixel)
std::cout << "RGB Values of central Pixel. RED value is: " << RED_Value << "GREEN value is: " << GREEN_Value << "BLUE value is: " << BLUE_Value <<std::endl;

/Reset Values
        RED_Value = 0;
        GREEN_Value = 0;
        BLUE_Value = 0;

//-----------------------------------------------------------------------------------------
//PIXEL FORCE
//-----------------------------------------------------------------------------------------
//Fix the central and 8 neighbours pixels to a constant value
        int RED_Value_New = 255;            //FIX your desired RED Value
        int GREEN_Value_New = 51;           //FIX your desired GREEN Value
        int BLUE_Value_New = 51;            //FIX your desired BLUE Value


        for(int i = row_central; i <= row_central; i++)
        {
                for(int j = col_central; j <= col_central; j++)
                {
                        //Copy Pixel to Variable
                        RED_Value = image.at<cv::Vec3b>(i, j)[0] ;
                        GREEN_Value = image.at<cv::Vec3b>(i, j)[1] ;
                        BLUE_Value = image.at<cv::Vec3b>(i, j)[2] ;

                        //Force Pixel Value
                        image.at<cv::Vec3b>(i, j)[0] = RED_Value_New;
                        image.at<cv::Vec3b>(i, j)[1] = GREEN_Value_New;
                        image.at<cv::Vec3b>(i, j)[2] = BLUE_Value_New;
                }
        }
//-----------------------------------------------------------------------------------------
    //show image in a window
    cv::imshow("Output Window", image);

        //print image dimensions
        //std::cout << "image size is: " << image.rows << "x" << image.cols << std::endl;

                //Waits 1 millisecond to check if a key has been pressed. If so, breaks the loop. Otherwise
continues.
    if(cv::waitKey(1) >= 0) break;
  }
}

//-----------------------------------------------------------------------------------------
//Show the reference Doc viewed to do the modifications of the base code
//-----------------------------------------------------------------------------------------
//REFERENCE DOC:
//doc: http://docs.opencv.org/doc/user_guide/ug_mat.html,
//http://stackoverflow.com/questions/8932893/accessing-certain-pixel-rgb-value-in-opencv
//http://stackoverflow.com/questions/7899108/opencv-get-pixel-channel-value-from-mat-image
//http://stackoverflow.com/questions/23001512/c-and-opencv-get-and-set-pixel-color-to-mat
//http://www.rapidtables.com/web/color/RGB_Color.htm
```